AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

53. (Currently amended) A computing environment having a plurality of computer components, each of the computer components being suitable for running a process which comprises ~~of~~ a plurality of objects each comprising data and program modules operating on the data, and which has an execution state at any time, the computing environment supporting:

transmittal of a process running on ~~any~~ a first one of the computer components to ~~any~~ a second one of the computer components to resume running on the second computer component, and

evolution of the process by selective deletion of ~~objects~~ part of the data and/or program modules and execution states from within the process and/or the selective addition of ~~objects~~ new data and/or program modules and execution states into the process and/or the selective replacement of ~~objects~~ part of the data and/or program modules and execution states from within the process by corresponding new ~~objects~~ data and/or program modules and execution states, thereby changing the functionality of the process.

54. (Currently amended) A computing environment as claimed in claim 53, wherein a construct is formed comprising data and/or program modules and execution state of a first process, and wherein said evolutionary operations are performed by functions operating on a said construct.

55. (Previously presented) A computing environment as claimed in claim 54, wherein said construct is formed by a construct operation that suspends all active threads of said first process and creates a new process comprising at least some of the data and/or program modules and execution state of said first process, and stores said new process in a data area of said first process.

56. (Previously presented) A computing environment as claimed in claim 55, wherein said construct comprises only data, program modules and execution state falling within lists that are passed to said construct operation.

57. (Previously presented) A computing environment as claimed in claim 54, wherein said construct is provided with an authorizing signature.

58. (Previously presented) A computing environment as claimed in claim 53, wherein said evolutionary operations include the selective deletion of objects from within said process.

59. (Previously presented) A computing environment as claimed in claim 53, wherein said evolutionary operations include the selective loading or reloading of objects into said process.

60. (Currently amended) A computing environment as claimed in claim 53, wherein ~~a~~ said evolutionary operation includes the incorporation into a first process of new objects from a second process.

61. (Previously presented) A computing environment as claimed in claim 60, wherein a construct is formed comprising at least some of the data and/or program modules and execution state from said second process, and said construct is transferred to said first process.

62. (Currently amended) A computing environment as claimed in claim 61, wherein said construct is formed by a construct operation that suspends all active threads

of said second process and creates a new process comprising a subset of the data, program modules and execution ~~sate~~ state of said second process, and stores said new process in a data area of said second process.

63. (Currently amended) A computing environment as claimed in claim 62, wherein said construct comprises only data, program modules and the execution state falling within lists that are passed to said construct operation.

64. (Previously presented) A computing environment as claimed in claim 61, wherein said construct is provided with an authorizing signature.

65. (Previously presented) A computing environment as claimed in claim 61, wherein after said construct is transferred, the second process stored within said construct is caused to be activated within said first process.

66. (Previously presented) A computing environment as claimed in claim 61, wherein after said construct is transferred the first process is suspended and the second process stored within said construct is activated, and when the second process is

concluded, the data and program modules of the second process are added to the first process and the first process is reactivated.

67. (Previously presented) A computing environment as claimed in claim 61, wherein after said first process terminates, at least some of the data and/or program modules from said first process are added to the second process stored in said construct and said second process is then activated.

68. (Previously presented) A computing environment as claimed in claim 60, wherein a construct is formed comprising at least some of the data and/or program modules from said second process, and said construct is transferred to said first process.

69. (Previously presented) A computing environment as claimed in claim 68, wherein said construct is formed by a construct operation that suspends all active threads of said second process and creates a new process comprising at least some of the data and/or program modules of said second process, and stores said new process in a data area of said second process.

70. (Previously presented) A computing environment as claimed in claim 69, wherein said construct comprises only a subset of data and program modules falling within lists that are passed to said construct operation.

71. (Previously presented) A computing environment as claimed in claim 68, wherein said data and said program modules from said second process are copied into said first process.

72. (Previously presented) A computing environment as claimed in claim 60, wherein in the event of a conflict between data and/or program modules of said first process and data and/or program modules of second process, the data and/or program modules of said first process will override the data and/or program modules of said second process.

73. (Previously presented) A computing environment as claimed in claim 60, wherein in the event of a conflict between data and/or program modules of said first process and data and/or program modules of said second process, the data and/or program modules of said second process will override the data and/or program modules of said first process.

74. (Currently amended) A computing environment as claimed in claim 53, wherein ~~a~~ said process may be transferred between different first and second hardware components of said computing environment.

75. (Previously presented) A computing environment as claimed in claim 74, wherein a construct is formed comprising at least some of the data and/or program modules and execution state of said process, and said construct is transferred.

76. (Previously presented) A computing environment as claimed in claim 74, wherein said construct comprises a subset of the data, program modules and execution state of said process.

77. (Currently amended) A computing environment as claimed in claim 74, wherein ~~a~~ said process is subject to an evolutionary operation that allows the process to run in the second hardware component.

78. (Previously presented) A computing environment as claimed in claim 74, wherein said second hardware component is a memory stage device.

79. (Currently amended) A method of operating a computer process within a computing environment defined by a plurality of computer components, the process comprising a plurality of objects which each comprise data and program modules operating on the data, the process running on a first of the computer components and having an execution state at any time, the method comprising:

transmitting the process to a second of the computer components, the process resuming operation on the second computer component, and

modifying the process by selective deletion of ~~objects~~ part of the data and/or program modules and execution states from within the process and/or the selective addition of ~~objects~~ new data and/or program modules and execution states into the process and/or the selective replacement of ~~objects~~ part of the data and/or program modules and execution states from within the process by corresponding new ~~objects~~ data and/or program modules and execution states thereby changing the functionality of the process.

80. (New) A computing environment according to claim 53, wherein evolution of the process to change its functionality is performed prior to the process resuming operation on the second computer component.

81. (New) A computing environment having a plurality of computer components, each of the computer components being suitable for running a process which comprises of plurality of objects each comprising data and program modules operating on the data, and which has an execution state at any time, the computing environment supporting:

transmittal of a process running on a first one of the computer components to a second one of the computer components to resume operation on the second computer component, and

prior to the process resuming operation on the second computer component, evolution of the process by selective deletion of objects from within the process and/or the selective addition of new objects into the process and/or the selective replacement of objects from within the process by new objects, thereby changing the functionality of the process.

82. (New) A method of operating a computer process within a computing environment defined by a plurality of computer components, the process comprising a plurality of objects which each comprise data and program modules operating on the data, the process running on a first of the computer components and having an execution state at any time, the method comprising:

transmitting the process to a second of the computer components,

modifying the process by selective deletion of objects from within the process and/or the selective addition of new objects into the process and/or the selective replacement of objects from within the process by new objects, thereby changing the functionality of the process, and thereafter

resuming operation of the said process on the second computer component.